MAIL ROOM JAN 13 1997 PAT. & TRADEMARK OFF. 21

ow. w 1AU

1-28-97
B. Hilliard
10/5

Patent
Attorney's Docket No. P1630:275

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of)
)
Max McFARLAND) Group Art Unit: 2415
)
Application No.: 08/435,375) Examiner: ~~B. Huynh~~
)
Filed: May 5, 1995)
)
For: SYSTEMS AND METHODS FOR REPLACING OPEN WINDOWS IN A GRAPHICAL USER INTERFACE)

RECEIVED
97 JAN 24 AM 7:54
GROUP 240

APPLICANT'S BRIEF PURSUANT TO 37 C.F.R. § 1.192(a)

Honorable Commissioner of Patents and Trademarks
Washington, D.C. 20231

Sir:

Further to the Notice of Appeal filed November 12, 1996 and in accordance with 37 C.F.R. § 1.192, Applicant respectfully submits the following brief in triplicate for the above-identified application.

A check covering the $300.00 requisite government fee is submitted herewith. However, the Commissioner is authorized to charge any fees that may be required by this paper, and to credit any overpayment, to Deposit Account No. 02-4800.

330 CL 01/23/97 08435375
1 120 300.00 CK

(1) Real Party in Interest

The assignee of the above-identified application is Apple Computer, Inc., a California corporation.

(2) Related Appeals and Interferences

Applicant is unaware of any other appeals or interferences which will directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

(3) Status of the Claims

Claims 1-7 are currently pending and all of these claims are on appeal. Claims 1-7 stand rejected under 35 U.S.C. § 103 as allegedly being unpatentable over Bates et al. (U.S. Patent No. 5,377,317).

(4) Status of Amendments

No Amendments have been filed after the Final Rejection dated July 9, 1996.

(5) Summary of the Invention

Applicant's invention relates to, for example, user interfaces which are provided to enable a user to interact with a computer. In particular, the present invention relates to the display of "windows" as part of the user interface. As seen in Figure 1 of this application, windows can be used as containers for icons that represent different functions or data. Some of the icons held within a window may be organizers which contain other icons, e.g., the folder icons seen in Figure 1. When the user activates such an icon, e.g., by "double-clicking" on the icon using a mouse, that icon then opens into a window of its own which is displayed in an overlapping fashion with the original window.

As a result, it is not unusual for a user to encounter a situation such as that illustrated in Figure 2 where a number (in this example five) of overlapping windows are displayed as part of the user interface. If the user then performs an operation which activates a window which is buried in the overlapping stack, that window will "spring" to a new location on the interface, e.g., as window 3 in Figure 3. When the user has completed her or his manipulation of objects within window 3, the window is then returned to the overlapping stack.

Conventionally, a returned window was simply placed on the top of the stack, as seen in Figure 4. However, Applicant recognized that this window placement was undesirable because it was not easy for a user to remember the placement of windows on the interface (particularly when there were many open windows on the interface at the same time) and because the returned window obscured the title bars of the other windows which remained in their original, overlapping position.

Thus, according to the present invention, a window is returned to its original placement in an overlapping stack after a springing operation has concluded. This can be accomplished, for example, by generating a list including the order of windows in an overlapping stack and then referring to this list when returning the window to the stack.

(6) Issues

The only issue on appeal is whether claims 1-7 are unpatentable under 35 U.S.C. § 103 over Bates et al. (U.S. Patent No. 5,377,317, hereinafter "Bates").

(7) Grouping of Claims

For the purposes of this appeal, each of the claims 1, 2, 3, 4 and 6 are submitted to be separately patentable and are argued separately below. Claim 5 stands or falls with claim 4 and claim 7 stands or falls with claim 6.

(8) Arguments

(a) Only Hindsight Usage of Applicant's Specification Suggests Returning Windows to Their Original Position

The Examiner has agreed that the window display order list of Bates is a dynamic variable computed on the basis of the extent to which a user has previously used a window, and thus fails to return a window to its original position as set forth, among other features, in Applicant's claim 1, for example. However, the Examiner asserts that it would have been obvious for one of ordinary skill in the art, at the time the invention was made, upon reading Bates, to implement a window list ordering that is not time variant, and thus return a window to its original position. The Examiner explains the motivation for modifying Bates in this way as "to return the windows to the locations which are already familiar to the users", which notion cannot be found by the undersigned in the Bates document. Instead, Bates suggests that users would prefer windows to be ordered based upon their relative activity time, e.g., windows which are used more frequently would cover less frequently used windows.

Thus, Applicant respectfully submits that the described motivation for modifying Bates in the manner set forth in the Office Action does not establish a *prima facie* case of obviousness. This is true because the motivation to "return the windows to the locations which are already familiar to the users" can only have blossomed from reading Applicant's specification, since it is clearly not a teaching or suggestion found in the Bates document or any other document of record.

It is noted that the Examiner is basing this ground of rejection on the general proposition that the elimination of an element, when its function is not desired, would have been obvious. This general proposition, however, does not extend to the selective and surgical removal of a small part of a device when such an elimination is only motivated by Applicant's specification, and particularly when the eliminated function is part of the essence of the described technique.

In this situation, the Examiner proposes exactly such a modification. Activity based ordering of windows is central to Bates' description. While it is true that Bates provides the capability to turn off his activity based ordering scheme to allow "conventional" window ordering (see column 6, lines 11-13 of Bates), the lengthy and repetitive discussion of activity-related ordering makes it exceedingly unlikely that one of ordinary skill in the art would have been motivated, upon reading only Bates, to implement a completely different scheme.

(b) <u>Neither Resetting nor Deactivating the Window Timing Function of Bates Would Have Resulted in Applicant's Claimed Combinations</u>

Even assuming, strictly *arguendo*, that it would have been obvious to modify Bates to implement a window list ordering that is not time variant as suggested in the Office Action, Applicant's claimed combinations still would not have resulted therefrom.

Referencing column 4, lines 29-34 of the Bates patent, the Examiner points out that Bates describes a window display order list that is generated by the computer. The order of window display is dictated by an amount of time that a user interacts with each of the given windows being displayed. The Examiner contends that the teaching of Bates suggests that if window timing is reset after every section [sic], or is permanently deactivated, then the window display order list will remain permanently unchanged and the windows will be displayed at the same position every time the system is started [sic]. Applicant respectfully disagrees with the Examiner's characterizations of Bates regarding the consequences of resetting and deactivating window timing for at least the following reasons.

A careful review of the Bates patent reveals that the windows will not necessarily be returned to an original position when Bates' window timing is reset or deactivated. For example, deactivation of the Bates system is described in the patent text at column 8, lines 54-64, which text refers to process step reference numbers set out in Figure 5C. Referring to Figure 5C, if a turn window timing off event is indicated (i.e., deactivation), an ON/OFF control flag is set to OFF at process step 166. If there are any window records at step 168, the process loops through step 169 where In focus and Total time data are set to zero for each open window. Since the system is turned off, it can only be assumed that event flags are not generally set, and that the process will flow through steps 170, 175 and 195 before being returned to general window processing through flow connectors F, D and B. As described in Bates at column 7, lines 36-38, following a determination that the ON/OFF control flag is set to OFF, the window timing program ends immediately at block 399. Because the program is shut off, the window reordering function is no longer available to a user. Thus, deactivating the window timing function of Bates would merely have the effect that no window reordering was performed.

With respect to the consequences of the resetting function of Bates, it appears that the Examiner has taken the position that column 6 of Bates teaches that by resetting the window timing, subsequent reordering of the windows will cause the windows to be redrawn in an original order. Applicant notes that the reset function of Bates is intended to be used, for example, when a user is performing a completely unrelated task relative to earlier performed tasks which provides an opportunity for the window timing to "start over". *See* col. 6, lines 29-35. Bates does not suggest using the reset function whenever the windows are to be reordered, as is apparently alleged by the Examiner.

Moreover, Applicant respectfully submits that by resetting the window timing, the window ordering in Bates will merely continue to be performed in accordance with an active timing indicator recorded for each of the open windows. That is, resetting the timing for the windows <u>does not reset the windows themselves</u>, but only resets the window active *timing* record ("in focus timing") to zero. Thus, Bates simply does not teach or suggest Applicant's claimed step of returning said window to said original position based upon said list, as set forth, among other steps, in claim 1.

(c) <u>Bates Lacks Other Claimed Elements In Addition To Those Admitted by the Examiner</u>

In the Final Office Action, the Examiner correctly admits that Bates fails to teach returning a window to its original position. The Examiner asserts that it would have been obvious to remedy this deficiency of Bates, which assertion is challenged above. However, other elements of the claimed combinations are also lacking in Bates, which deficiencies are not addressed in the Final Office Action.

Claim 1

Claim 1 recites, inter alia, generating a list which provides a front-to-back order of said plurality of cascaded windows and an indicator of whether each of said plurality of cascaded windows is currently in its respective original, cascaded position. This feature of Applicant's claim 1 combination is not taught or suggested by Bates.

In the Office Action dated January 18, 1996 (the substance of which is incorporated by reference into the Final Office Action at paragraph 3), the Examiner analogizes window list 27 seen in Figure 2D of Bates with the above-identified claimed list. The Examiner asserts that "[t]his list provides an indication of whether or not a window is in its original position." Applicant challenges this assertion.

Window list 27 is a list which can be displayed on the user interface. As described at column 5, lines 6-16 of Bates, this list is simply a list of window titles ordered by amount of activity. The user can scroll through this list to select a new window for activation. **<u>However, nowhere does Bates mention that this list includes an indicator of whether each window is currently in its original, cascaded position as claimed</u>**. Nor would there be any need to provide this information in window list 27, because this list functions as a shortcut for a user to retrieve a window and the user wouldn't care about its current position vis-a-vis its original position.

Claim 2

Claim 2 includes, inter alia, comparing an identifier of said window with an identifier associated with each window in said list until a match occurs; and placing said window behind a window which is next in order in said list after said match occurs. These steps are not taught or suggested by Bates.

In the original Office Action, the Examiner notes that Bates does not teach these steps but, now taking for granted that Bates teaches returning a window to its original position, asserts that the step of comparing must somehow be implicit in the teachings of Bates. This bootstrapping argument is simply not supported by the evidence relied upon by the Examiner. Moreover, note that the window list 27 does not even include the claimed identifier with which to make a comparision.

Claim 3

Claim 3 includes, inter alia, placing said window behind said window which is next in order in said list after said match occurs only if said window is currently in its respective original, cascaded position. This step is clearly not taught or suggested by Bates. Nowhere in Bates is there any disclosure of tracking whether a window is currently in its respective original, cascaded position, much less the operation of claim 3 which relies upon this information.

Claim 4

Claim 4 recites, inter alia, the steps of generating a list which indicates that said first window is to be rendered behind said second window when both said first and said second windows are rendered in said first portion of said display space and placing said first window behind said second window in said first portion of said display space by making reference to said list. These steps are also not taught or suggested by Bates.

Again, Bates does not tie together the rendering of specific windows relative to one another when they are both rendered in a portion of the display space by way of a list. Instead, Bates relies upon the relative activity of windows to determine their display status.

Claim 6

Claim 6 recites, inter alia, a data structure for storing information associated with said window object and said at least one other window object including a relative time-invariant order of said window object with respect to said at least one other window object. This feature is not taught or suggested by Bates.

As mentioned above, Bates teaches ordering windows based upon an activity level associated with each window. Thus Bates expressly teaches away from the provision of a data structure including a time invariant order of the window with respect to another window.

(9) Conclusion

In order to determine whether a prima facie case of obviousness under 35 U.S.C. § 103 exists, it is necessary to ascertain whether the prior art teachings would appear to be sufficient to one of ordinary skill in the art to suggest making the claimed substitution or other modification. In re Lalu, 747 F.2d 703, 705, 223 U.S.P.Q. 1257, 1258 (Fed. Cir. 1984).

Since the applied document fails to establish such a prima facie case for at least the foregoing reasons, it is respectfully requested that the Examiner's rejection of the claims as being unpatentable under 35 U.S.C. § 103 over Bates be REVERSED.

Respectfully submitted,

BURNS, DOANE, SWECKER & MATHIS, L.L.P.

By [illegible]
Steven M. du Bois
Registration No. 35,023

699 Prince Street
P.O. Box 1404
Alexandria, VA 22313-1404
Phone No. (703) 836-6620

Dated: January 13, 1997

APPENDIX

1. A method for returning a window to an original position among a plurality of cascaded windows which are rendered on a display space, comprising the steps of:

generating a list which provides a front-to-back order of said plurality of cascaded windows and an indicator of whether each of said plurality of cascaded windows is currently in its respective original, cascaded position;

removing said window from said original position;

rendering said window at another location on said display space;

receiving, at a graphical interface, an indication that said window is to be removed from said another location on said display space; and

returning said window to said original position based upon said list generated by said step of generating.

2. The method of claim 1, wherein said step of returning further comprises the steps of:

comparing an identifier of said window with an identifier associated with each window in said list until a match occurs; and

placing said window behind a window which is next in order in said list after said match occurs.

3. The method of claim 2, wherein said step of placing said window further comprises the step of:

placing said window behind said window which is next in order in said list after said match occurs only if said window is currently in its respective original, cascaded position.

4. A method for placing a first window behind a second window in a first portion of a display space, comprising the steps of:

generating a list which indicates that said first window is to be rendered behind said second window when both said first and said second windows are rendered in said first portion of said display space;

removing said first window from behind said second window;

rendering said first window at a second portion of said display space;

removing said first open window from said second portion of said display space; and

placing said first window behind said second window in said first portion of said display space by making reference to said list.

5. The method of claim 4, wherein a third window is disposed in front of both said second window and said first window when rendered in said first portion of said display space, said method further comprising the steps of:

removing said second window from said first portion of said display space;

rendering said second window in another portion of said display space; and

placing said first window behind said third window in said first portion of said display space.

6. In a computer having a display, a system for returning a window object to its original location relative to at least one other window object, comprising:

a data structure for storing information associated with said window object and said at least one other window object including a relative time-invariant order of said window object with respect to said at least one other window object;

a display on which said window object and said at least one other window object are rendered;

a graphical user interface for receiving and generating signals associated with said window object and said at least one other window object, including a signal indicating that said window object is to be returned to said original position; and

a processor for receiving said signal and drawing said window object on said display at said original position using said information in said data structure.

7. The system of claim 6, wherein said data structure also include information indicating a position of said at least one other window object, and wherein said processor selectively draws said window object in an overlapping relationship with said other window object based upon said position indicating information in said data structure.